Query Answering in DL-Lite with Datatypes: A Non-Uniform Approach

André Hernich and **Julio Lemos** and **Frank Wolter**
University of Liverpool
Department of Computer Science
{hernich, jlemos, wolter}@liverpool.ac.uk

Abstract

Adding datatypes to ontology-mediated queries (OMQs) often makes query answering hard. As a consequence, the use of datatypes in OWL 2 QL has been severely restricted. In this paper we propose a new, non-uniform, way of analyzing the data-complexity of OMQ answering with datatypes. Instead of restricting the ontology language we aim at a classification of the patterns of datatype atoms in OMQs into those that can occur in non-tractable OMQs and those that only occur in tractable OMQs. To this end we establish a close link between OMQ answering with datatypes and constraint satisfaction problems over the datatypes. In a case study we apply this link to prove a P/coNP-dichotomy for OMQs over DL-Lite extended with the datatype $(\mathbb{Q}, \leq)$. The proof employs a recent dichotomy result by Bodirsky and Kára for temporal constraint satisfaction problems.

1 Introduction

In recent years, querying data using ontologies has become one of the main applications of description logics (DLs). The general idea is that an ontology is used to enrich incomplete and heterogenous data with a semantics and with background knowledge and thereby serves as an interface for querying data that also allows the derivation of additional facts. In this area called ontology-based data management (OBDM) one of the main research problems is to identify ontology languages and queries for which query answering scales to large amounts of data (Calvanese et al. 2007; Bienvenu and Ortiz 2015). In DL, ontologies take the form of a TBox, data is stored in an ABox, and the most important class of queries are (unions of) conjunctive queries, or simply (U)CQs. A basic observation regarding this setup is that even for DLs from the DL-Lite family that have been designed for tractable OBDM the addition of datatypes to the TBoxes or the UCQs easily leads to non-tractable query answering problems (Artale, Ryzhikov, and Kontchakov 2012; Savkovic and Calvanese 2012). As a consequence of this, the use of datatypes in TBoxes and query languages for OBDM has been severely restricted (Motik and Horrocks 2008; Motik et al. 2009). In applications, however, there is clearly a need for expressive datatypes both in TBoxes and in queries.

The aim of this paper is to revisit OBDM with expressive datatypes from a new, non-uniform perspective. Instead of the standard approach that aims at the definition of DLs $\mathcal{L}$ and query languages $\mathcal{Q}$ such that for any TBox $\mathcal{T}$ in $\mathcal{L}$ and any query q in $\mathcal{Q}$, answering q under $\mathcal{T}$ is tractable in data-complexity we now aim at describing the complexity of query answering with datatypes at a more fine-grained level by taking into account the way in which datatype atoms can occur in TBoxes and in queries. To this end, we establish a close link between the complexity of query answering and of constraint satisfaction problems (CSPs) over the datatype. This link enables us to transfer complexity results from the CSP world to the world of OBDM and leads, in some cases, to complete classifications of the complexity of query answering into PTime and coNP-complete classes.

In more detail, we consider TBoxes in the DL DL-Lite$_\mathcal{R}$ underpinning the OWL profile OWL 2 QL extended with concept inclusions that contain attribute restrictions qualified by unary datatype atoms on their right hand side and UCQs that contain datatype atoms of arbitrary arity. If $\mathcal{T}$ is such a TBox over datatype $\mathcal{D}$ and q such a UCQ over $\mathcal{D}$, then $Q = (\mathcal{T}, q)$ is called an ontology-mediated query (OMQ) over $\mathcal{D}$. We aim at understanding the complexity of query answering for this very broad class of OMQs. A first observation is that query answering becomes undecidable for many important datatypes $\mathcal{D}$ including $(\mathbb{Q}, <)$, $(\mathbb{Z}, <)$ and $(\mathbb{Z}, \leq)$. To restore decidability and enable a polynomial reduction to the complement of CSPs over $\mathcal{D}$ we introduce the bounded match depth property (BMDP), a new property of OMQs that ensures that answers to OMQs can be determined based on a bounded subset of the standard chase of a DL-Lite$_\mathcal{R}$ knowledge base. This property is a generalization of the bounded derivation depth property in (Calì, Gottlob, and Lukasiewicz 2012). Many practical OMQs have the BMDP. For example, all OMQs with either TBoxes whose chase always terminates (which is often the case in practice (Grau et al. 2013)) or with rooted UCQs whose variables are all connected via non-datatype variables to answer variable (which covers a broad class of UCQs). If the datatype $\mathcal{D}$ is homogeneous (as is the case for $(\mathbb{Q}, <)$ and $(\mathbb{Q}, \leq)$), then the latter condition can be relaxed even further to certain Boolean UCQs. As the CSP of many important datatypes is in NP, it follows that query answering for OMQs with the BMDP over such datatypes is in coNP, a significant im-

provement compared to the undecidable OMQs without the BMDP. To sharpen the link between OMQ answering and CSP further, we also provide a converse polynomial reduction of CSPs over a datatype $\mathcal{D}$ to the complement of answering OMQs with BMDP over $\mathcal{D}$. This converse reduction can thus be used to transfer NP-hardness results from the CSP world to coNP-hardness results for OMQ answering. More importantly, however, we now have a framework for transfering *complexity classification results* from CSP to OMQ answering.

We illustrate the power of this framework for the datatype $(\mathbb{Q}, \leq)$. Note first that even without qualified attribute restrictions OMQs over datatype $(\mathbb{Q}, \leq)$ can express many interesting queries.

Example 1.1. Let $\mathcal{T} = \{\exists p \sqsubseteq \exists U, \exists p^- \sqsubseteq \exists U\}$ be a TBox, where p is a role name and U an attribute. Then the Boolean CQ

$$q \leftarrow p(x, y) \wedge U(x, u) \wedge U(y, v) \wedge u \leq v$$

is entailed by a KB $(\mathcal{T}, \mathcal{A})$ over $(\mathbb{Q}, \leq)$ such that $\mathcal{A}$ is an ABox containing no assertions using U iff $\mathcal{A}$ contains a p-cycle. Thus, answering the OMQ $(\mathcal{T}, q)$ is NLogSpace-complete. We also construct OMQs over $(\mathbb{Q}, \leq)$ that are PTime-complete and, respectively, coNP-complete.

Our main result is a P/coNP-dichotomy for OMQs over $(\mathbb{Q}, \leq)$ with the BMDP. To formulate the dichotomy we associate with every OMQ $Q = (\mathcal{T}, q)$ a datatype pattern $\mathsf{dtype}(Q) = (\theta_\mathcal{T}, \theta_q)$ such that $\theta_\mathcal{T}$ contains the datatype atoms in $\mathcal{T}$ and θ_q contains the datatype atoms in q. In Example 1.1, $\theta_\mathcal{T} = \emptyset$ and $\theta_q = \{u \leq v\}$. Then, based on the framework introduced above and a recent P/NP-dichotomy result for temporal CSPs (Bodirsky and Kára 2010a) we show that for any datatype pattern θ exactly one of the following two conditions holds (unless P=coNP):

- Evaluating OMQs Q with BMDP and $\mathsf{dtype}(Q) = \theta$ is always in PTime.
- There exists an OMQ Q with BMDP and $\mathsf{dtype}(Q) = \theta$ whose evaluation problem is coNP-hard.

In addition, our dichotomy comes with a purely syntactic description of the datatype patterns that lead to OMQs that are in PTime. For example, the datatype pattern in Example 1.1 will always lead to an OMQ in PTime.

Related Work Expressive DLs with datatypes (or concrete domains) have been introduced in (Baader and Hanschke 1991) and studied extensively (Lutz 2002). In the context of tractable DLs, reasoning with datatypes has been studied in (Baader, Brandt, and Lutz 2005; Magka, Kazakov, and Horrocks 2011) for $\mathcal{EL}$ and in (Poggi et al. 2008; Savkovic and Calvanese 2012; Artale, Ryzhikov, and Kontchakov 2012) for DL-Lite. These works focus on finding ontology languages for which typical reasoning tasks are tractable. In contrast, here we start with ontology languages for which query answering is intractable in general, and aim at a complexity classification of query answering guided by the datatype pattern. Our methodology is closely related to recent work relating OBDM to constraint satisfaction problems (Lutz and Wolter 2012; Bienvenu et al. 2014; Hernich et al. 2015). However, here we classify datatype patterns according to the data-complexity of evaluating the OMQs containing them, whereas in (Lutz and Wolter 2012) TBoxes are classified according to the data-complexity of OMQs containing them, and in (Bienvenu et al. 2014) OMQs themselves are classified according to their data-complexity. Consequently, here we establish a link to temporal constraint satisfaction (Bodirsky and Kára 2010a) whereas the work mentioned above establishes a link to standard constraint satisfaction and the Feder-Vardi conjecture (Feder and Vardi 1993; Bulatov, Jeavons, and Krokhin 2005).

Detailed proofs are provided in the full version of this paper.

2 Preliminaries

A *datatype* is a tuple $\mathcal{D} = (D, R_1, R_2, \dots)$, where D is a non-empty set and $R_1, R_2, \dots$ are relations on D. We call $\mathrm{dom}(\mathcal{D}) := D$ the *domain* of $\mathcal{D}$. To simplify the presentation, we will not distinguish between a relation R_i and its name (i.e., we use R_i both as a relation and as the name of the relation R_i).

A *primitive positive (PP) formula* over $\mathcal{D}$ is a first-order formula φ built from atomic formulas over $\mathcal{D}$ using conjunction and existential quantification. If φ has no free variables, then φ is called a *PP sentence* over $\mathcal{D}$. By $\mathsf{CSP}(\mathcal{D})$ we denote the problem of deciding whether a given PP sentence over $\mathcal{D}$ is satisfied in $\mathcal{D}$. PP formulas over $\mathcal{D}$ do not use domain elements of $\mathcal{D}$ as constants. If we admit such constants we speak about PP formulas or sentences over $\mathcal{D}$ *with constants* and denote the satisfiction problem for such PP sentences by $\mathsf{CSP}_c(\mathcal{D})$.

We assume countably infinite and mutually disjoint sets of *concept names*, *role names*, *attribute names*, and *individual names*. *Roles* r and *basic concepts* B are defined by the rules

$$r := p \mid p^- \quad B := A \mid \exists r \mid \exists U$$

where p ranges over all role names, A ranges over all concept names, and U ranges over all attribute names. A *DL-Lite*$_\mathcal{R}^{attrib}(\mathcal{D})$ *TBox* is a finite set of *role inclusions* $r_1 \sqsubseteq r_2$, *attribute inclusions* $U_1 \sqsubseteq U_2$, and *concept inclusions* $B_1 \sqsubseteq B_2$ and $B_1 \sqsubseteq \neg B_2$, where r_1, r_2 range over roles, U_1, U_2 over attributes, and B_1, B_2 over basic concepts. In TBoxes of the extension *DL-Lite*$_\mathcal{R}^{qattrib}(\mathcal{D})$ of *DL-Lite*$_\mathcal{R}^{attrib}(\mathcal{D})$ one can use, in addition, *qualified attribute restrictions* on the right hand side of inclusions:

$$B \sqsubseteq \exists U.\varphi, \quad B \sqsubseteq \forall U.\varphi$$

where B is a basic concept, U is an attribute, and φ is a PP formula over $\mathcal{D}$ with constants and a single free variable x.

Let $\mathcal{D}$ be a datatype. A *$\mathcal{D}$-ABox* consists of *assertions* of the form $A(a)$, $p(a, b)$, and $U(a, u)$, where A is a concept name, p is a role name, U is an attribute name, a, b are individual names, and $u \in \mathrm{dom}(\mathcal{D})$. A *$\mathcal{D}$-knowledge base ($\mathcal{D}$-KB)* is a pair $(\mathcal{T}, \mathcal{A})$ consisting of a *DL-Lite*$_\mathcal{R}^{qattrib}(\mathcal{D})$ TBox $\mathcal{T}$ and a $\mathcal{D}$-ABox $\mathcal{A}$.

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ over a datatype $\mathcal{D}$ consists of a non-empty domain $\Delta^{\mathcal{I}} = \Delta^{\mathcal{I}}_{\mathsf{ind}} \cup \mathrm{dom}(\mathcal{D})$ and an interpretation function $\cdot^{\mathcal{I}}$ that assigns to each concept name A a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathsf{ind}}$, to each role name p a relation $p^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathsf{ind}} \times \Delta^{\mathcal{I}}_{\mathsf{ind}}$, and to each attribute name U a relation $U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}_{\mathsf{ind}} \times \mathrm{dom}(\mathcal{D})$. The elements in $\Delta^{\mathcal{I}}_{\mathsf{ind}}$ are called *individuals*, whereas the elements in $\mathrm{dom}(\mathcal{D})$ are called *data values*. We assume that $\Delta^{\mathcal{I}}_{\mathsf{ind}}$ and $\mathrm{dom}(\mathcal{D})$ are disjoint. Throughout this paper, we make the *standard name assumption*: if $\mathcal{I}$ is an interpretation, then we set $a^{\mathcal{I}} := a$ for all individual names a. We also set $u^{\mathcal{I}} := u$ for each $u \in \mathrm{dom}(\mathcal{D})$, and $R^{\mathcal{I}} := R$ for each relation R of $\mathcal{D}$. The interpretation $\mathcal{I}$ induces the interpretations $B^{\mathcal{I}}$ and $r^{\mathcal{I}}$ for each basic concept B and role r in the standard way (Artale et al. 2009). The qualified attribute restrictions are interpreted as follows:

$$(\exists U.\varphi)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}}_{\mathsf{ind}} \mid \exists s\, ((a,s) \in U^{\mathcal{I}} \,\&\, \mathcal{D} \models \varphi(s))\},$$
$$(\forall U.\varphi)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}}_{\mathsf{ind}} \mid \forall s\, ((a,s) \in U^{\mathcal{I}} \Rightarrow \mathcal{D} \models \varphi(s))\}.$$

The interpretation $\mathcal{I}$ is called *model* of a $\mathcal{D}$-ABox $\mathcal{A}$ if $a \in A^{\mathcal{I}}$, $(a,b) \in p^{\mathcal{I}}$, and $(a,u) \in U^{\mathcal{I}}$ for all assertions $A(a)$, $p(a,b)$, and $U(a,u)$ in $\mathcal{A}$. It is called *model* of a *DL-Lite*$^{qattrib}_{\mathcal{R}}(\mathcal{D})$ TBox $\mathcal{T}$ if $X^{\mathcal{I}} \subseteq Y^{\mathcal{I}}$ for every inclusion $X \sqsubseteq Y \in \mathcal{T}$.

An interpretation is a model of a $\mathcal{D}$-KB $(\mathcal{T}, \mathcal{A})$ if it is a model of both $\mathcal{A}$ and $\mathcal{T}$. A $\mathcal{D}$-KB $(\mathcal{T}, \mathcal{A})$ is *satisfiable* if it has a model; in this case we say that $\mathcal{A}$ is *satisfiable relative to* $\mathcal{T}$.

Conjunctive queries (CQs) q over $\mathcal{D}$ take the form $q(\bar{x}) \leftarrow \varphi$, where $\bar{x}$ is the tuple of *answer variables* of q, and φ is a conjunction of atomic formulas of the form $A(y)$, $p(y,z)$, $U(y,u)$, or $R(u_1,\ldots,u_k)$, where A, p, U, and R range over concept names, role names, attribute names, and relation names in $\mathcal{D}$, respectively; each y, z and each $u, u_1, \ldots, u_k$ is a *variable* and the variables $u, u_1, \ldots, u_k$ are called *data variables*. As usual, all variables of $\bar{x}$ must occur in some atom of φ. A *match* of q in an interpretation $\mathcal{I}$ is a mapping μ from the variables of φ to $\Delta^{\mathcal{I}}$ such that for each atom $X(t_1,\ldots,t_k)$ of φ we have $(\mu(t_1),\ldots,\mu(t_k)) \in X^{\mathcal{I}}$. A tuple $\bar{c}$ of individual names and data values is an *answer* to q in an interpretation $\mathcal{I}$ if there is a match μ of q in $\mathcal{I}$ such that $\mu(\bar{x}) = \bar{c}$. We denote this by $\mathcal{I} \models q(\bar{c})$.

A *union of conjunctive queries (UCQ)* q over $\mathcal{D}$ takes the form $q_1(\bar{x}), \ldots, q_n(\bar{x})$, where each $q_i(\bar{x})$ is a CQ over $\mathcal{D}$. The q_i are called *disjuncts* of q. A tuple $\bar{c}$ of individual names and data values is an *answer* to q in an interpretation $\mathcal{I}$, denoted by $\mathcal{I} \models q(\bar{c})$, if $\bar{c}$ is an answer to some disjunct of q in $\mathcal{I}$.

Given a $\mathcal{D}$-KB $(\mathcal{T}, \mathcal{A})$, a UCQ q over $\mathcal{D}$, and a tuple $\bar{c}$ of individual names and data values, we write $\mathcal{T}, \mathcal{A} \models q(\bar{c})$ if $\bar{c}$ is an answer to q in every model of $(\mathcal{T}, \mathcal{A})$.

An *ontology-mediated query (OMQ)* over $\mathcal{D}$ takes the form $Q = (\mathcal{T}, q)$, where $\mathcal{T}$ is a *DL-Lite*$^{qattrib}_{\mathcal{R}}(\mathcal{D})$ TBox and q is a UCQ over $\mathcal{D}$. Given a $\mathcal{D}$-ABox $\mathcal{A}$ and a tuple $\bar{c}$, we write $\mathcal{A} \models Q(\bar{c})$ if $\mathcal{T}, \mathcal{A} \models q(\bar{c})$.

When studying the OMQ answering problem we focus on data complexity, unless otherwise stated.

3 Universal Pre-Models

Many ontology languages for which query evaluation is tractable admit the construction of universal models of KBs using some variant of the chase procedure. This applies to DL-Lite$_{\mathcal{R}}$ as well. In contrast, KBs in its extension *DL-Lite*$^{qattrib}_{\mathcal{R}}(\mathcal{D})$ often do not have universal models and the aim of this section is to introduce instead models with placeholders for data values that play a similar role as universal models, but modulo the assignment of data values to the placeholders. The following examples illustrates why universal models do not always exist.

Example 3.1. Consider the KB $(\mathcal{T}, \mathcal{A})$ over the datatype $(\mathbb{Q}, \leq)$ with $\mathcal{T} = \{A \sqsubseteq \exists U_1, A \sqsubseteq \exists U_2\}$. and $\mathcal{A} = \{A(a)\}$. Consider the OMQs $Q_i = (\mathcal{T}, q_i)$, for $i \in \{1,2\}$, where

$$q_1(x) \leftarrow U_1(x,u_1) \wedge U_2(x,u_2) \wedge u_1 \leq u_2,$$
$$q_2(x) \leftarrow U_1(x,u_1) \wedge U_2(x,u_2) \wedge u_2 \leq u_1.$$

Clearly $\mathcal{A} \not\models Q_1(a)$ since for the interpretation $\mathcal{I}$ with $U_1^{\mathcal{I}} = \{(a,2)\}$ and $U_2^{\mathcal{I}} = \{(a,1)\}$ we have $\mathcal{I} \not\models q_1(a)$. Also, $\mathcal{A} \not\models Q_2(a)$ since for the interpretation $\mathcal{J}$ with $U_1^{\mathcal{J}} = \{(a,1)\}$ and $U_2^{\mathcal{J}} = \{(a,2)\}$ we have $\mathcal{J} \not\models q_2(a)$. However, there does not exist a model $\mathcal{I}$ of $\mathcal{T}$ and $\mathcal{A}$ such that $\mathcal{I} \not\models q_i(a)$ for both $i = 1$ and $i = 2$.

The reason that universal models do not exist is that distinct interpretations of attributes can be required to refute the entailment of CQs or UCQs. The notion of pre-interpretation formalizes this intuition: it fixes the interpretation of concept and roles names but leaves the interpretation of attributes open by adding placeholders for data values (called data nulls) to the set of possible values of attributes.

Fix a $\mathcal{D}$-KB $(\mathcal{T}, \mathcal{A})$. A *pre-interpretation* $\mathcal{J}$ over $\mathcal{D}$ is the same as an interpretation over $\mathcal{D}$ with the exception that attribute names U are now interpreted as sets $U^{\mathcal{J}} \subseteq \Delta^{\mathcal{J}}_{\mathsf{ind}} \times (\mathrm{dom}(\mathcal{D}) \cup \Delta^{\mathcal{J}}_{\mathsf{null}})$, where $\Delta^{\mathcal{J}}_{\mathsf{null}}$ is a set of *data nulls* disjoint from $\Delta^{\mathcal{J}}_{\mathsf{ind}} \cup \mathrm{dom}(\mathcal{D})$, and that for each $u \in \Delta^{\mathcal{J}}_{\mathsf{null}}$ we fix a set $Z^{\mathcal{J}}(u)$ of PP formulas φ that occur in qualified attribute restrictions of $\mathcal{T}$. Intuitively, the formulas in $Z^{\mathcal{J}}(u)$ restrict the possible data values in $\mathrm{dom}(\mathcal{D})$ that can be assigned to u. The definitions of the interpretations $C^{\mathcal{J}}$ of a concept C and $S^{\mathcal{J}}$ of a role or attribute S are extended from interpretations to pre-interpretations in the straightforward way. A *pre-model* of a KB is a pre-interpretation that satisfies all assertions and inclusions in the KB.

Pre-interpretations $\mathcal{J}$ can be completed to interpretations by assigning data values to data nulls. A *completion function* f for $\mathcal{J}$ is a mapping $f\colon \Delta^{\mathcal{J}}_{\mathsf{null}} \to \mathrm{dom}(\mathcal{D})$ such that for all $u \in \Delta^{\mathcal{J}}_{\mathsf{null}}$ and $\varphi \in Z^{\mathcal{J}}(u)$ we have $\mathcal{D} \models \varphi(f(u))$. The *completion* $f(\mathcal{J})$ of $\mathcal{J}$ by f is the interpretation $\mathcal{I}$ obtained from $\mathcal{J}$ by replacing each data null u in $\mathcal{J}$ by $f(u)$, and by dropping the sets $Z^{\mathcal{J}}(u)$.

Using a straightforward modification of the standard chase procedure for *DL-Lite*$_{\mathcal{R}}$ (see, e.g., (Calvanese et al. 2007)) one can construct a pre-model $\mathrm{can}(\mathcal{T}, \mathcal{A})$ of any satisfiable $\mathcal{D}$-KB $(\mathcal{T}, \mathcal{A})$ such that for any UCQ q over $\mathcal{D}$ and

any $\bar{c}$:

$$(\mathcal{T},\mathcal{A}) \models q(\bar{c}) \iff f(\text{can}(\mathcal{T},\mathcal{A})) \models q(\bar{c}),$$
for all completion functions f.

We call $\text{can}(\mathcal{T},\mathcal{A})$ with this property a *universal pre-model* of $\mathcal{T}$ and $\mathcal{A}$.

Lemma 3.2. *For every satisfable $\mathcal{D}$-KB $(\mathcal{T},\mathcal{A})$ there exists a universal pre-model can$(\mathcal{T},\mathcal{A})$ of $\mathcal{T}$ and $\mathcal{A}$.*

The following example illustrates the construction of universal pre-models.

Example 3.3. Extend the TBox $\mathcal{T}$ from Example 3.1 to $\mathcal{T}'$ by adding $A \sqsubseteq \forall U_1.x \geq 0$ and $A \sqsubseteq \forall U_1.x \leq 1$. The universal pre-model $\text{can}(\mathcal{T}',\mathcal{A})$ is given by setting $\Delta^{\text{can}(\mathcal{T},\mathcal{A})} = \{a,u_1,u_2\}$; $A^{\text{can}(\mathcal{T},\mathcal{A})} = \{a\}$; $U_1^{\text{can}(\mathcal{T},\mathcal{A})} = \{(a,u_1)\}$; $U_2^{\text{can}(\mathcal{T},\mathcal{A})} = \{(a,u_2)\}$; $Z^{\text{can}(\mathcal{T},\mathcal{A})}(u_1) = \{0 \leq x,\, x \leq 1\}$, and $Z^{\text{can}(\mathcal{T},\mathcal{A})}(u_2) = \emptyset$. A completion function f for $\text{can}(\mathcal{T},\mathcal{A})$ maps u_1 to a rational number $f(u_1)$ with $0 \leq f(u_1) \leq 1$ and u_2 to some rational number and defines a completion $f(\text{can}(\mathcal{T},\mathcal{A}))$ in which U_1 is interpreted as $(a, f(u_1))$ and U_2 is interpreted as $(a, f(u_2))$.

4 Query Evaluation and CSP

Universal pre-models can be infinite and their completions can have a very complicated structure. In fact, we begin this section with the observation that there are many important and simple datatypes $\mathcal{D}$ such that evaluating OMQs over $\mathcal{D}$ is undecidable, and this holds even for OMQs with TBoxes without qualified attribute restrictions. We then introduce the bounded match depth property (BMDP) of OMQs which entails that evaluating OMQs can only be undecidable if a CSP over $\mathcal{D}$ is undecidable. Even more importantly, the BMDP allows us to establish mutual polynomial reductions between evaluating OMQs over a datatype $\mathcal{D}$ and CSPs over $\mathcal{D}$. We also establish the BMDP for a large class of OMQs.

Theorem 4.1. *Let $\mathcal{D} \in \{(\mathbb{Z},\neq),(\mathbb{Z},<),(\mathbb{Z},\leq),(\mathbb{Q},\neq),(\mathbb{Q},<)\}$. Then the query evaluation problem for OMQs over $\mathcal{D}$ with DL-Lite$_{\mathcal{R}}^{attrib}(\mathcal{D})$ TBoxes is undecidable in combined complexity.*

For $(\mathbb{Z},\neq)$ and $(\mathbb{Q},\neq)$ the proof is by reduction of the $\mathbb{N}\times\mathbb{N}$ tiling problem and very similar to known undecidability proofs for query evaluation with either UCQs with inequality or TBoxes with key constraints (Toman and Weddell 2008; Rosati 2007; Gutiérrez-Basulto et al. 2015). As $x \neq y$ iff $x < y$ or $y < x$, undecidability of query evaluation with inequality in the datatype entails undecidability of query evaluation with $<$ in the datatype. Finally, undecidability of query evaluation for $(\mathbb{Z},\leq)$ can be proved by reduction of query evaluation for $(\mathbb{Z},<)$. It remains open whether query evaluation is decidable for OMQs over $(\mathbb{Q},\leq)$ and whether undecidability holds for data complexity.

The proof of Theorem 4.1 makes heavy use of TBoxes whose chase does not terminate and of UCQs that are not *safe* or *rooted*. Given a CQ q over $\mathcal{D}$, we call any two variables x,y *connected* in q if x and y are connected via a path of atoms in q that does not contain data variables. Then q is *rooted* if any non-data variable is connected to a non-data answer variable and q is *safe* if any two non-data variables of q are either connected in q or are both connected to non-data answer variables. A UCQ is rooted (safe) if all its disjuncts are rooted (and safe, respectively) and an OMQ is rooted (safe) if its UCQ is rooted (and safe, respectively). Thus, every rooted query is safe, but the converse does not hold. Note that safe queries do not allow us to compare attribute values of individuals that are arbitrarily far apart in $\text{can}(\mathcal{T},\mathcal{A})$.

Example 4.2. The CQ in Example 1.1 is safe but not rooted, the CQs in Example 3.1 are rooted. In contrast, the Boolean CQ given by $q \leftarrow U_1(x_1,u_1) \wedge U_2(x_2,u_2) \wedge u_1 \leq u_2$ is not safe.

An important shared property of all OMQs with TBoxes with a terminating chase, all rooted OMQs, and all safe OMQs over homogeneous datatypes[1] is that they can be answered on a finite initial portion of $\text{can}(\mathcal{T},\mathcal{A})$ whose size only depends on the OMQ. Given an integer $d \geq 0$, let $\text{can}^d(\mathcal{T},\mathcal{A})$ be the subinterpretation of $\text{can}(\mathcal{T},\mathcal{A})$ induced by the set of domain elements that are reachable from ABox elements in at most d steps. An OMQ $Q = (\mathcal{T},q)$ over datatype $\mathcal{D}$ enjoys the *bounded match depth property (BMDP)* if there exists a $d \geq 0$ such that for all $\mathcal{D}$-ABoxes $\mathcal{A}$ and all tuples $\bar{c}$,

$$\mathcal{T},\mathcal{A} \models q(\bar{c}) \iff f(\text{can}^d(\mathcal{T},\mathcal{A})) \models q(\bar{c}), \qquad (\star)$$
for all completion functions f.

This property closely resembles the bounded derivation depth property in (Calì, Gottlob, and Lukasiewicz 2012), with the key difference being that it crucially depends on the notion of completion.

It is not difficult to verify that all OMQs with a TBox with a terminating chase and all rooted OMQs $(\mathcal{T},q)$ have the BMDP. In fact, in the latter case $(\star)$ holds when d is the maximum number of atoms in a disjunct of q. A much more sophisticated argument shows that safe OMQs over homogeneous datatypes enjoy the BMDP.

Lemma 4.3. *The following OMQs enjoy the BMDP: safe OMQs over homogeneous datatypes; rooted OMQs; and OMQs using a TBox $\mathcal{T}$ such that $\text{can}(\mathcal{T},\mathcal{A})$ is finite for each ABox $\mathcal{A}$.*

We now show that every OMQ $Q = (\mathcal{T},q)$ over $\mathcal{D}$ that enjoys the BMDP can be reduced to a constraint satisfaction problem $\mathsf{CSP}_c(\Gamma)$ whose constraint language Γ is defined solely by the patterns of formulas over $\mathcal{D}$ that occur in Q. We describe these patterns using the notion of the *datatype pattern* of Q. The *datatype pattern* of Q is defined as $\mathsf{dtype}(Q) = (\theta_{\mathcal{T}},\theta_q)$, where $\theta_{\mathcal{T}}$ is the set of all PP formulas that occur in qualified attribute restrictions of $\mathcal{T}$, and θ_q contains, for each disjunct q' of q, the conjunction of all atoms of q' over $\mathcal{D}$. Note that $\theta_{\mathcal{T}}$ is a set of PP formulas with constants and a single free variable, whereas θ_q is a set of PP formulas without constants. We refer to datatype patterns of OMQs over $\mathcal{D}$ as *datatype patterns over $\mathcal{D}$*.

[1] A datatype $\mathcal{D}$ is homogeneous if every isomorphism between finite induced substructures extends to an automorphism on $\mathcal{D}$. $(\mathbb{Q},<)$ and $(\mathbb{Q},\leq)$ are examples of homogeneous datatypes (Chang and Keisler 1998).

Example 4.4. Let $\mathcal{T} = \{A \sqsubseteq \exists U_1.1 \leq x \wedge x \leq 3\}$ be a $\mathcal{D}$-TBox, for $\mathcal{D} = (\mathbb{Q}, \leq)$, and let q be the UCQ over $\mathcal{D}$ given by $q_1(x), q_2(x)$, where

$$q_1(x) \leftarrow \textstyle\bigwedge_{i=1}^{3} U_i(x, z_i) \wedge z_1 \leq z_2 \wedge z_1 \leq z_3,$$
$$q_2(x) \leftarrow U_1(x, z_1') \wedge U_2(x, z_2') \wedge z_2' \leq z_1'.$$

Then $\mathsf{dtype}(\mathcal{T}, q) = (\theta_\mathcal{T}, \theta_q)$, where $\theta_\mathcal{T} = \{1 \leq x \wedge x \leq 3\}$ and $\theta_q = \{z_1 \leq z_2 \wedge z_1 \leq z_3, z_2' \leq z_1'\}$.

Now, with each datatype pattern $\theta = (\theta_\mathcal{T}, \theta_q)$ over $\mathcal{D}$, where $\theta_\mathcal{T} = \{\varphi_1, \ldots, \varphi_m\}$ and $\theta_q = \{\psi_1, \ldots, \psi_n\}$, we associate the constraint language

$$\Gamma_\theta = (\mathrm{dom}(\mathcal{D}), R_{\varphi_1}, \ldots, R_{\varphi_m}, R_{\neg\psi_1}, \ldots, R_{\neg\psi_n}),$$

where for each formula $\varphi(x_1, \ldots, x_k)$ over $\mathcal{D}$, we let $R_\varphi = \{\bar{a} \in \mathrm{dom}(\mathcal{D})^k \mid \mathcal{D} \models \varphi(\bar{a})\}$.

Theorem 4.5. *Let θ be a datatype pattern over $\mathcal{D}$.*

If $Q = (\mathcal{T}, q)$ is an OMQ over $\mathcal{D}$ with $\mathsf{dtype}(Q) = \theta$ and Q enjoys the BMDP, then evaluating Q is polynomially reducible to the complement of $\mathsf{CSP}_c(\Gamma_\theta)$.

Conversely, there is a rooted OMQ Q over $\mathcal{D}$ with $\mathsf{dtype}(Q) = \theta$ such that the complement of $\mathsf{CSP}_c(\Gamma_\theta)$ is polynomially reducible to evaluating Q.

Proof. Let $\theta = (\theta_\mathcal{T}, \theta_q)$, where $\theta_\mathcal{T} = \{\varphi_1, \ldots, \varphi_m\}$ and $\theta_q = \{\psi_1, \ldots, \psi_n\}$. Assume Q enjoys the BMDP, and that q is given as $q_1(\bar{x}), \ldots, q_n(\bar{x})$. Let $\mathcal{A}$ be a $\mathcal{D}$-ABox and let $\bar{c}$ be a tuple of individual names and data values of the same length as $\bar{x}$. It is not difficult to see that satisfiability of $\mathcal{D}$-ABoxes $\mathcal{A}$ relative to $\mathcal{T}$ is polynomially reducible to $\mathsf{CSP}_c(\mathrm{dom}(\mathcal{D}), R_{\varphi_1}, \ldots, R_{\varphi_m})$. Thus, we can assume that $\mathcal{A}$ is satisfiable relative to $\mathcal{T}$. Consider the pre-model $\mathcal{I} := \mathrm{can}^d(\mathcal{T}, \mathcal{A})$, where d is an integer that satisfies $(\star)$ but is independent of $\mathcal{A}$. A *pre-match* of q_i in $\mathcal{I}$ is a match of the abstract part of q_i (i.e. q_i stripped off of all atoms over $\mathcal{D}$) in $\mathcal{I}$. Let X_i be the set of all pre-matches μ of q_i in $\mathcal{I}$ with $\mu(\bar{x}) = \bar{c}$, and let $\bar{u} = u_1, \ldots, u_k$ be a repetition-free enumeration of all the data nulls that occur in the image of some pre-match in $X_1 \cup \cdots \cup X_n$. Then the following is an instance of $\mathsf{CSP}_c(\Gamma_\theta)$:

$$\Phi := \exists \bar{u} \left(\bigwedge_{i=1}^{k} \bigwedge_{\varphi \in Z^\mathcal{I}(u_i)} R_\varphi(u_i) \wedge \bigwedge_{i=1}^{n} \bigwedge_{\mu \in X_i} R_{\neg\psi_i}(\mu(\bar{z}_i)) \right),$$

where $u_1, \ldots, u_k$ are identified with individual variables in Φ. It is easy to check that $\mathcal{T}, \mathcal{A} \not\models q(\bar{c})$ iff $\Gamma_\theta \models \Phi$.

For the converse, we encode each instance Φ of $\mathsf{CSP}_c(\Gamma_\theta)$ by an ABox $\mathcal{A}_\Phi$ as follows. We use a distinguished individual a_Φ to denote the root of $\mathcal{A}_\Phi$. For each atom $\alpha = R_{\neg\psi_i}(x_1, \ldots, x_k)$ in Φ there is an individual b_α connected to a_Φ via an assertion $r_i(a_\Phi, b_\alpha)$. For each $j \in \{1, \ldots, k\}$, this individual is connected to individual c_{x_j} via an assertion $s_j(b_\alpha, c_{x_j})$. Finally, for each variable x that occurs in Φ we include the assertion $A(c_x)$; if $R_{\varphi_i}(x)$ occurs in Φ we additionally include $A_{\varphi_i}(c_x)$. Furthermore, for each element $u \in \mathrm{dom}(\mathcal{D})$ that occurs in Φ we include the assertion $U(c_u, u)$. Figure 1 illustrates this construction for the case $m = 1$, $n = 3$, and Φ being

$$\exists x, y \, (R_{\varphi_1}(x) \wedge R_{\neg\psi_1}(0, x) \wedge R_{\neg\psi_1}(x, y) \wedge R_{\neg\psi_3}(y, 1, x)) \, .$$

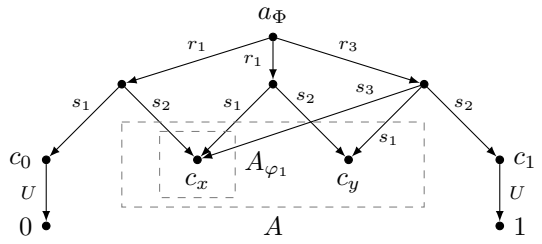

Figure 1: ABox $\mathcal{A}_\Phi$ from the proof of Theorem 4.5

Let $\mathcal{T} = \{A \sqsubseteq \exists U\} \cup \{A_{\varphi_i} \sqsubseteq \forall U.\varphi_i \mid 1 \leq i \leq m\}$ and let q be the UCQ given by $q_1(x), \ldots, q_n(x)$, where $q_i(x)$ is

$$r_i(x, y) \wedge \bigwedge_{1 \leq j \leq k} (s_j(y, z_j) \wedge U(z_j, u_j)) \wedge \psi_i(u_1, \ldots, u_k)$$

and k is the number of free variables of ψ_i. Clearly, $Q = (\mathcal{T}, q)$ is a rooted OMQ over $\mathcal{D}$ with $\mathsf{dtype}(Q) = \theta$. It is straightforward to verify that $\Gamma_\theta \models \Phi$ if and only if $\mathcal{T}, \mathcal{A}_\Phi \not\models q(a_\Phi)$. □

Note that the TBoxes constructed in the converse direction of the proof of Theorem 4.5 are very simple. In particular, the chase terminates on any given ABox.

It follows immediately from Theorem 4.5 that for the datatypes $\mathcal{D}$ given in Theorem 4.1 the query evaluation problem for OMQs with the BMDP over $\mathcal{D}$ is in coNP. In the next section we show how Theorem 4.5 can be used to understand the OMQs with the BMDP whose query evaluation problem is not only in coNP but in PTime.

5 The Datatype $(\mathbb{Q}, \leq)$

The results of the previous section establish a tight link between evaluating OMQs and solving CSPs. In particular, they allow us to transfer complexity classification results from CSP to OMQ answering. We now demonstrate the power of this link for the datatype $(\mathbb{Q}, \leq)$. Our main result is the following P/coNP-dichotomy of evaluating OMQs over $(\mathbb{Q}, \leq)$ that enjoy the BMDP based on their datatype patterns. To simplify the presentation, we assume w.l.o.g. that for all datatype patterns $\theta = (\theta_\mathcal{T}, \theta_q)$ over $(\mathbb{Q}, \leq)$ the formulas in θ_q are acyclic.[2]

We use *min-pattern* and *max-pattern* to refer to formulas of the form $x_0 \leq x_1 \wedge \cdots \wedge x_0 \leq x_k$ and $x_1 \leq x_0 \wedge \cdots \wedge x_k \leq x_0$, for $k \geq 0$, respectively.

Theorem 5.1. *Let $\theta = (\theta_\mathcal{T}, \theta_q)$ be a datatype pattern over $(\mathbb{Q}, \leq)$, where $\theta_q = \{\psi_1, \ldots, \psi_n\}$.*

If each ψ_i is a min-pattern or each ψ_i is a max-pattern, then evaluating OMQs Q over $(\mathbb{Q}, \leq)$ with $\mathsf{dtype}(Q) = \theta$ and the BMDP is in PTime.

Otherwise, there is a rooted OMQ Q over $(\mathbb{Q}, \leq)$ with $\mathsf{dtype}(Q) = \theta$ such that evaluating Q is coNP-complete.

[2] Cycles $x_1 \leq x_2 \leq \cdots \leq x_n \leq x_1$ that occur in a formula in θ_q can always be eliminated by removing all their atoms and replacing each x_i with x_1.

The simple and purely syntactic characterization of the tractable cases of Theorem 5.1 makes it very easy to verify whether evaluating a given OMQ over $(\mathbb{Q}, \leq)$ with the BMDP is guaranteed to be tractable or possibly coNP-complete. For instance, Theorem 5.1 implies that the OMQ $(\mathcal{T}, q)$ in Example 4.4 and in general all OMQs over $(\mathbb{Q}, \leq)$ that have the same datatype pattern and enjoy the BMDP can be evaluated in PTime. On the other hand, if we consider the datatype pattern that has $z_1 \leq z_2 \wedge z_2 \leq z_3$ in place of $z_1 \leq z_2 \wedge z_1 \leq z_3$, then there are rooted OMQs over $(\mathbb{Q}, \leq)$ with that datatype pattern for which evaluation is coNP-complete.

We now take a closer look at the proof of Theorem 5.1. The main device is a recent dichotomy for *temporal CSPs* by Bodirsky and Kára (2010a). Temporal CSPs are defined as $\mathsf{CSP}(\Gamma)$, where $\Gamma = (\mathbb{Q}, R_1, R_2, \dots)$ and each R_i is definable by a first-order formula $\Phi_i(\bar{x})$ over $(\mathbb{Q}, <)$ without constants, i.e., $R_i = \{\bar{a} \mid (\mathbb{Q}, <) \models \Phi_i(\bar{a})\}$. The datatypes Γ of a temporal CSP are called *temporal constraint languages*. Bodirsky and Kára (2010a) prove that for every temporal constraint language Γ, $\mathsf{CSP}(\Gamma)$ is either in PTime or NP-complete. Whether or not a given temporal constraint language defines a tractable CSP depends only on which functions preserve its relations. Bodirsky and Kára consider a set $\mathcal{F}$ of five types of functions $f\colon \mathbb{Q}^2 \to \mathbb{Q}$ together with the set *dual-*$\mathcal{F}$ consisting of the duals *dual-*$f(x, y) := -f(-x, -y)$ of all functions f in $\mathcal{F}$.[3] A function $f \in \mathcal{F} \cup$ *dual-*$\mathcal{F}$ *preserves* a relation $R \subseteq \mathbb{Q}^n$ if for all $(a_1, \dots, a_n), (b_1, \dots, b_n) \in R$ we have $(f(a_1, b_1), \dots, f(a_n, b_n)) \in R$. We also say that f preserves a temporal constraint language Γ if f preserves all relations in Γ.

Theorem 5.2. (Bodirsky and Kára 2010a) *Let Γ be a temporal constraint language. If Γ is preserved by a function in $\mathcal{F} \cup$ dual-$\mathcal{F}$, then $\mathsf{CSP}(\Gamma)$ is in PTime. Otherwise, $\mathsf{CSP}(\Gamma)$ is NP-complete.*

Together with Theorem 4.5 and a constant-elimination technique, this leads to a basic P/coNP-dichotomy for evaluating OMQs Q over $(\mathbb{Q}, \leq)$ with the BMDP based on the preservation properties of $\Gamma_{\mathsf{dtype}(Q)}$.

Theorem 5.3. *Let $\theta = (\theta_{\mathcal{T}}, \theta_q)$ be a datatype pattern over $(\mathbb{Q}, \leq)$, where $\theta_q = \{\psi_1, \dots, \psi_n\}$.*

If there is a function in $\mathcal{F} \cup$dual-$\mathcal{F}$ that preserves $R_{\neg\psi_i}$ for each $i \in \{1, \dots, n\}$, then evaluating OMQs Q over $(\mathbb{Q}, \leq)$ with $\mathsf{dtype}(Q) = \theta$ and the BMDP is in PTime.

Otherwise, there is a rooted OMQ Q over $(\mathbb{Q}, \leq)$ with $\mathsf{dtype}(Q) = \theta$ such that evaluating Q is coNP-complete.

To obtain a purely syntactic characterization of the datatype patterns $\theta = (\theta_{\mathcal{T}}, \theta_q)$ over $(\mathbb{Q}, \leq)$ that lead to tractable OMQs, we further analyze the relations $R_{\neg\psi}$ that are defined by the formulas $\psi \in \theta_q$ and are preserved under some function in $\mathcal{F} \cup$ *dual-*$\mathcal{F}$. Note that the negation $\neg\psi$ of each formula $\psi \in \theta_q$ is equivalent to a disjunction Ψ of atomic formulas $x < y$, with x and y variables. Moreover, since ψ is acyclic (by assumption), the directed graph with the variables of Ψ as its vertices, and edges (y, x) for each atomic formula $x < y$ of Ψ is acyclic. We call such formulas Ψ *acyclic disjunctive* formulas. The following lemma is the combinatorial core of our analysis.

Lemma 5.4. *Let $R \subseteq \mathbb{Q}^n$ be defined by an acyclic disjunctive formula Ψ over $(\mathbb{Q}, <)$. If R is preserved by a function in $\mathcal{F} \cup$ dual-$\mathcal{F}$, then Ψ has the form $\bigvee_{i=1}^{k} x_i < x_0$ if $f \in \mathcal{F}$, and $\bigvee_{i=1}^{k} x_0 < x_i$ if $f \in$ dual-$\mathcal{F}$.*

Altogether, this leads to a proof of Theorem 5.1.

Proof of Theorem 5.1. By Theorem 5.3, it suffices to show that the following are equivalent:

1. Each $\psi \in \theta_q$ is a min-pattern, or each $\psi \in \theta_q$ is a max-pattern.
2. There is a function $f \in \mathcal{F} \cup$ *dual-*$\mathcal{F}$ such that each $R_{\neg\psi}$, $\psi \in \theta_q$, is preserved under f.

If each $\psi \in \theta_q$ is a min-pattern, then for each $\psi \in \theta_q$ the relation $R_{\neg\psi}$ is defined by a formula of the form $\bigvee_{i=1}^{n} x_0 > x_i$. Similarly, if each $\psi \in \theta_q$ is a max-pattern, then for each $\psi \in \theta_q$ the relation $R_{\neg\psi}$ is defined by a formula of the form $\bigvee_{i=1}^{n} x_i > x_0$. It is known (Bodirsky and Kára 2010b, Proposition 3.5) that relations defined by such formulas are preserved under a function in $\mathcal{F} \cup$ *dual-*$\mathcal{F}$.

Conversely, let f be a function in $\mathcal{F} \cup$ *dual-*$\mathcal{F}$ such that each $R_{\neg\psi}$ with $\psi \in \theta_q$ is preserved under f. Let $\psi \in \theta_q$. Then, $R_{\neg\psi}$ is defined by an acyclic disjunctive formula Ψ. By Lemma 5.4, Ψ has the form $\bigvee_{i=1}^{n} x_i < x_0$, if $f \in \mathcal{F}$, and $\bigvee_{i=1}^{n} x_0 < x_i$, if $f \in$ *dual-*$\mathcal{F}$. This implies that each $\psi \in \theta_q$ is a min-pattern, or each $\psi \in \theta_q$ is a max-pattern. □

We now refine the analysis of the datatype patterns that lead to OMQs with an evaluation problem in PTime further by presenting a dichotomy between those that can be used in PTime-hard OMQs and those that always lead to OMQs in NLogSpace. Example 1.1 shows that there are rooted OMQs Q over $(\mathbb{Q}, \leq)$ and with $\mathsf{dtype}(Q) = (\emptyset, \{x \leq y\})$ such that evaluating Q is NLogSpace-complete. It turns out that the NLogSpace upper bound holds for all OMQs Q whose datatype pattern contains atomic formulas only.

Theorem 5.5. *Evaluating OMQs Q over $(\mathbb{Q}, \leq)$ with the BMDP and $\mathsf{dtype}(Q) = (\theta_{\mathcal{T}}, \theta_q)$ such that each formula in θ_q is of the form $x_0 \leq x_1$ is in NLogSpace.*

The proof of Theorem 5.5 is a straightforward application of Part 1 of Theorem 4.5, which allows us to reduce evaluating OMQs as in Theorem 5.5 to the complement of $\mathsf{CSP}_c(\mathbb{Q}, <, \leq)$ (it is not difficult to see that this reduction can be carried out in logarithmic space), and the observation that $\mathsf{CSP}_c(\mathbb{Q}, <, \leq)$ is in NLogSpace via a simple reachability test. The following result entails that the NLogSpace upper bound cannot be generalised to further tractable datatype patterns.

Theorem 5.6. *There is a rooted OMQ Q over $(\mathbb{Q}, \leq)$ with $\mathsf{dtype}(Q) = (\emptyset, \{x \leq y \wedge x \leq z\})$ such that evaluating Q is PTime-complete.*

The proof of Theorem 5.6 is by reduction of the alternating reachability problem.

[3] The set $\mathcal{F}$ consists of the functions min, mi, mx, and ll, defined in (Bodirsky and Kára 2010a), and all constant functions.

6 Conclusion

We have presented a framework for analyzing the non-uniform complexity of OMQ answering with expressive datatypes by establishing a close link to CSPs. We have illustrated the power of this framework by transfering a P/coNP dichotomy result for CSPs over $(\mathbb{Q}, \leq)$ to OMQ answering over $(\mathbb{Q}, \leq)$. Many research questions arise within this framework, including the following: (1) The framework should be applied to analyze the complexity of OMQ answering for other important datatypes based on the rationals, the integers, strings, or other structures used in spatial and temporal reasoning. On the CSP side there has been very significant progress in understanding the complexity of such structures (Bodirsky 2015). (2) We have established the BMDP for many relevant OMQs, but a more systematic investigation covering additional datatypes and TBoxes would be useful. (3) We have focused on establishing worst-case complexity results and dichotomies for OMQ answering. It would be of great interest to exploit the CSP reduction further and develop practical query answering algorithms, in particular, to use constraint solvers as part of practical query engines.

Acknowledgments. This work was supported by the EPSRC under the grant EP/M012646/1 "iTract: Islands of Tractability in Ontology-Based Data Access".

References

Artale, A.; Calvanese, D.; Kontchakov, R.; and Zakharyaschev, M. 2009. The *DL-Lite* family and relations. *J. Artif. Intell. Res. (JAIR)* 36:1–69.

Artale, A.; Ryzhikov, V.; and Kontchakov, R. 2012. *DL-Lite* with attributes and datatypes. In *ECAI*, 61–66.

Baader, F., and Hanschke, P. 1991. A scheme for integrating concrete domains into concept languages. In *IJCAI 1991*, 452–457.

Baader, F.; Brandt, S.; and Lutz, C. 2005. Pushing the $\mathcal{EL}$ envelope. In *IJCAI-05*, 364–369.

Bienvenu, M., and Ortiz, M. 2015. Ontology-mediated query answering with data-tractable description logics. In *Reasoning Web 2015*, 218–307.

Bienvenu, M.; ten Cate, B.; Lutz, C.; and Wolter, F. 2014. Ontology-based data access: A study through disjunctive datalog, csp, and MMSNP. *ACM Trans. Database Syst.* 39(4):33:1–33:44.

Bodirsky, M., and Kára, J. 2010a. The complexity of temporal constraint satisfaction problems. *J. ACM* 57(2):9:1–9:41.

Bodirsky, M., and Kára, J. 2010b. A fast algorithm and datalog inexpressibility for temporal reasoning. *ACM Trans. Comput. Log.* 11(3):15:1–15:21.

Bodirsky, M. 2015. The complexity of constraint satisfaction problems (invited talk). In *STACS 2015*, 2–9.

Bulatov, A. A.; Jeavons, P.; and Krokhin, A. A. 2005. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.* 34(3):720–742.

Calì, A.; Gottlob, G.; and Lukasiewicz, T. 2012. A general datalog-based framework for tractable query answering over ontologies. *J. Web Sem.* 14:57–83.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning* 39(3):385–429.

Chang, C., and Keisler, H. 1998. *Model Theory*. Elsevier.

Feder, T., and Vardi, M. Y. 1993. Monotone monadic SNP and constraint satisfaction. In *Proc. of the ACM Symposium on Theory of Computing*, 612–622.

Grau, B. C.; Horrocks, I.; Krötzsch, M.; Kupke, C.; Magka, D.; Motik, B.; and Wang, Z. 2013. Acyclicity notions for existential rules and their application to query answering in ontologies. *J. Artif. Intell. Res. (JAIR)* 47:741–808.

Gutiérrez-Basulto, V.; Ibáñez-García, Y. A.; Kontchakov, R.; and Kostylev, E. V. 2015. Queries with negation and inequalities over lightweight ontologies. *J. Web Sem.* 35:184–202.

Hernich, A.; Lutz, C.; Ozaki, A.; and Wolter, F. 2015. Schema.org as a description logic. In *IJCAI 2015*, 3048–3054.

Lutz, C., and Wolter, F. 2012. Non-uniform data complexity of query answering in description logics. In *KR 2012*.

Lutz, C. 2002. Description logics with concrete domains-a survey. In *Advances in Modal Logic 4*, 265–296.

Magka, D.; Kazakov, Y.; and Horrocks, I. 2011. Tractable extensions of the description logic $\mathcal{EL}$ with numerical datatypes. *J. Autom. Reasoning* 47(4):427–450.

Motik, B., and Horrocks, I. 2008. OWL datatypes: Design and implementation. In *ISWC 2008*, 307–322.

Motik, B.; Grau, B. C.; Horrocks, I.; Wu, Z.; Fokoue, A.; and Lutz, C. 2009. Owl 2 web ontology language: Profiles. World Wide Web Consortium, Working Draft WD-owl2-profiles-20081202.

Poggi, A.; Lembo, D.; Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2008. Linking data to ontologies. *J. Data Semantics* 10:133–173.

Rosati, R. 2007. The limits of querying ontologies. In Schwentick, T., and Suciu, D., eds., *Database Theory - ICDT 2007*, volume 4353 of *Lecture Notes in Computer Science*, 164–178. Springer.

Savkovic, O., and Calvanese, D. 2012. Introducing datatypes in *DL-Lite*. In *ECAI 2012*, 720–725.

Toman, D., and Weddell, G. E. 2008. On keys and functional dependencies as first-class citizens in description logics. *J. Autom. Reasoning* 40(2-3):117–132.